IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	David E. Lowell, et al.	§	Art Unit:	2191
		§		
Serial No.:	10/676,557	§	Confirmation No.:	7663
		§		
Filed:	October 1, 2003	§	Examiner:	Qing Chen
		§		
For:	Interposing a Virtual Machine Monitor and Devirtualizing Computer Hardware	§	Atty. Dkt. No.:	200208633-1 (HPC.0522US)

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

The final rejection of claims 1, 3-47, 49-56, and 58-74 is hereby appealed.

I. REAL PARTY IN INTEREST

The real party in interest is the Hewlett-Packard Development Company, LP. The Hewlett-Packard Development Company, LP, a limited partnership established under the laws of the State of Texas and having a principal place of business at 11445 Compaq Center Drive West, Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

II. RELATED APPEALS AND INTERFERENCES

The following appeals may be related to, directly affect, or be directly affected by, or have a bearing on the Board's position in the pending Appeal:

U.S. Serial No.	Status
10/677,159	Appeal Brief filed 05/17/2010 Notice of Appeal filed 03/15/2010 Appeal Brief filed 02/05/2009 Notice of Appeal filed 12/05/2008
10/676,922	Appeal Brief filed 04/26/2010 Notice of Appeal filed 02/25/2010 Appeal Brief filed 01/09/2009 Notice of Appeal filed 11/11/2008 Appeal Brief filed 05/27/2008 Notice of Appeal filed 03/26/2008

III. STATUS OF THE CLAIMS

Claims 1, 3-47, 49-56, and 58-74 have been finally rejected and are the subject of this appeal.

Claims 2, 48, and 57 have been cancelled.

IV. STATUS OF AMENDMENTS

An Amendment under 37 C.F.R. § 1.116 was filed on August 13, 2010 to address informalities raised in the Final Office Action. Entry of this Amendment is appropriate since the scope of the claims has not changed.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the independent claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element. Note also that the cited passages are provided as examples, as other passages in the specification or drawings not cited may also be relevant to the corresponding claim elements.

Independent claim 1 recites a method of using a virtual machine monitor and an operating system on computer hardware in a computer, the method comprising:

interposing (Fig. 2a:212) the virtual machine monitor (Fig. 1:116) between the computer hardware (Fig. 1:110) and the operating system (Fig. 1:112) at runtime, wherein the interposing occurs after booting of the computer, and wherein interposing the virtual machine monitor gives the virtual machine monitor direct control of at least a portion of the computer hardware (Spec., p. 4, ¶ [0019], ln. 1 - p. 5, ¶ [0022], ln. 4; p. 6, ¶ [0027], ln. 1-2); and

booting (Fig. 2a:210) the operating system on the computer hardware before interposing the virtual machine monitor at runtime (Spec., p. 4, ¶ [0020], ln. 2-6).

Independent claim 19 recites a method of using a virtual machine monitor and an operating system on virtualized computer hardware, the method comprising devirtualizing (Fig. 2a:214; Fig. 2b:254) the virtualized computer hardware at runtime of a computer containing the virtualized computer hardware, wherein runtime includes a period of execution in the computer after booting and before shutdown (Spec., p. 5, ¶ [0021], ln. 2-4; ¶ [0024], ln. 1-9; p. 6, ¶ [0025], ln. 4-6; p. 13, ¶ [0050], ln. 1-2),

wherein devirtualizing the virtualized computer hardware comprises stopping the virtual machine monitor (Spec., p. 14, ¶ [0052], ln. 1-3).

Independent claim 31 recites a computer comprising hardware (Fig. 1:110), the hardware including memory, the memory encoded with an operating system(Fig. 1:112), a virtual machine monitor (Fig. 1:116), and code for interposing (Fig. 2a:212) the virtual machine monitor between the operating system and the hardware at runtime, wherein the interposing occurs after booting of the computer (Spec., p. 4, ¶ [0019], ln. 1 - p. 5, ¶ [0022], ln. 4; p. 6, ¶ [0027], ln. 1-2),

wherein the operating system is to be booted (Fig. 2a:210) in the computer before interposing the virtual machine monitor (Spec., p. 4, ¶ [0020], ln. 2-6).

Independent claim 41 recites a computer comprising hardware (Fig. 1:110), the hardware including memory, the memory encoded with a virtual machine monitor (Fig. 1:116) to virtualize the hardware, and code for devirtualizing (Fig. 2a:214; Fig. 2b:254) the hardware at runtime, wherein runtime includes a period of execution in the computer after booting and before shutdown (Spec., p. 5, ¶ [0021], ln. 2-4; ¶ [0024], ln. 1-9; p. 6, ¶ [0025], ln. 4-6; p. 13, ¶ [0050], ln. 1-2), and wherein devirtualizing the hardware comprises stopping the virtual machine monitor (Spec., p. 14, ¶ [0052], ln. 1-3).

Independent claim 52 recites an article for use with an operating system on computer hardware, the article comprising a computer-readable storage medium storing software that when executed by the computer causes the computer to:

virtualize (Fig. 2a:212) at least a portion of the computer hardware (Fig. 1:110) at runtime by providing a virtual machine monitor (Fig. 1:116) between the operating system (Fig. 1:112) and the computer hardware, wherein the virtualizing occurs after booting of the computer and loading of the operating system (Spec., p. 4, ¶ [0019], ln. 1 - p. 5, ¶ [0022], ln. 4; p. 6, ¶ [0027], ln. 1-2), and

wherein the operating system is to be booted (Fig. 2a:210) in the computer before virtualizing the at least a portion of the computer hardware at runtime (Spec., p. 4, ¶ [0020], ln. 2-6).

Independent claim 61 recites an article for running an operating system (Fig. 1:112) and a virtual machine monitor (Fig. 1:116) on a computer, the computer including an I/O device (Fig. 1:110), the article comprising computer memory encoded with an I/O driver having first and second modes of operation (Spec., p. 10, ¶ [0041], ln. 1 - p. 11, ¶ [0042], ln. 14), the I/O driver operable (Fig. 6:612) in the first mode to interface directly between the operating system and the I/O device (Spec., p. 12, ¶ [0046], ln. 4-13), the I/O driver operable (Fig. 5:512) in the second mode to interface between the operating system and a corresponding I/O driver of the virtual machine monitor (Spec., p. 11, ¶ [0044], ln. 3-10).

Independent claim 62 recites an article for use with an operating system on computer hardware, the article comprising a computer-readable storage medium storing software that when executed by a computer causes the computer to devirtualize (Fig. 2a:214; Fig. 2b:254) at least a portion of virtualized hardware (Fig. 1:110) at runtime, wherein runtime is a period of execution in the computer after booting and before shutdown (Spec., p. 5, ¶ [0021], ln. 2-4; ¶ [0024], ln. 1-9; p. 6, ¶ [0025], ln. 4-6; p. 13, ¶ [0050], ln. 1-2), and wherein devirtualizing the at least a portion of the virtualized hardware comprises stopping a virtual machine monitor interposed between the operating system and the hardware (Spec., p. 14, ¶ [0052], ln. 1-3).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL[1]

A. Claims 1, 3-47, 49-56, 58-60, and 62-74 were rejected under 35 U.S.C. § 102(e) as anticipated by Nelson (U.S. Patent No. 6,961,941).

B. Claim 61 was rejected under 35 U.S.C. § 102(b) as anticipated by Bugnion (U.S. Patent No. 6,075,938).

VII. ARGUMENT

The claims do not stand or fall together. Instead, Appellant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-headings as required by 37 C.F.R. § 41.37(c)(1)(vii).

A. Claims 1, 3-47, 49-56, 58-60, and 62-74 were rejected under 35 U.S.C. § 102(e) as anticipated by Nelson (U.S. Patent No. 6,961,941).

1. Claims 1, 4-7, 9-16, 18.

It is respectfully submitted that claim 1 is clearly not anticipated by Nelson.

Claim 1 recites a method of using a virtual machine monitor and an operating system on computer hardware in a computer, the method comprising:

> interposing the virtual machine monitor between the computer hardware and the operating system at runtime, wherein the interposing occurs after booting of the computer, and wherein interposing the virtual machine monitor gives the virtual machine monitor direct control of at least a portion of the computer hardware; and
>
> booting the operating system on the computer hardware before interposing the virtual machine monitor at runtime.

Note that according to claim 1, the virtual machine monitor is **interposed** between the computer hardware and the operating system at runtime, wherein the interposing occurs after

[1] Claim objections and rejections under 35 U.S.C. § 112, ¶ 2, have been rendered moot by the Amendment under 37 C.F.R. § 1.116 filed on August 13, 2010.

booting the computer. Significantly, note also that the **operating system** of claim 1 is booted on the computer hardware **before interposing the virtual machine monitor at runtime**.

The rejection of claim 1 appears to be quite confused on what in Nelson is considered the "operating system" of claim 1. The main text of the rejection of claim 1 on pages 4 and 5 of the 03/17/2010 Office Action refers to both the commodity operating system (COS) of Nelson and the kernel of Nelson, as depicted in Fig. 1 of Nelson. As will be explained further below, neither the COS nor the kernel can properly be considered the "operating system" of claim 1.

The Response to Arguments section of the Office Action further pointed to the virtual OS (VOS) of Nelson as being the "operating system" of the claimed subject matter. *Id.* at 55. Note that the virtual operating system (VOS) in Nelson is part of the virtual machine (VM) 200 depicted in Fig. 1 of Nelson. As discussed in Nelson, the VM is installed to run on the kernel via the VMM. Nelson, 3:41-50. The VMM provides an interface between the VM and the underlying host operating system and hardware. *Id.*, 1:52-53. Therefore, it is clear that the VM is not installed until **after** the kernel (600) and VMM (300) depicted in Fig. 1 of Nelson have been installed. The kernel 600 and VMM 300 of are installed after the COS has initialized and booted the computer system. *Id.*, 3:5-16. Therefore, the VOS in the VM of Nelson cannot be the operating system of claim 1, because the VOS of Nelson is not booted on the computer hardware **before interposing the VMM at runtime**. The express words of claim 1 are as follows: "booting the operating system on the computer hardware **before** interposing the virtual machine monitor at runtime." The VOS of Nelson is installed in the VM, which in turn is installed **after** the VMM has been interposed in Nelson. Therefore, contrary to the allegation made in the Response to Arguments section of the Office Action, the VOS of Nelson cannot be the operating system of claim 1.

As noted above, the main text of the rejection of claim 1 on pages 4 and 5 also refer to the COS and the kernel of Nelson that are depicted in Fig. 1 of Nelson. Neither the COS nor the kernel of Nelson can satisfy the "operating system" of claim 1.

If the COS (420) shown in Fig. 1 of Nelson is considered the "operating system" of claim 1, then it is clear that the VMM 300 also shown in Fig. 1 of Nelson is **not** interposed between the COS 420 and the hardware. Instead, the VMM is provided between a virtual machine 200 and the kernel 600, as shown in Fig. 1 of Nelson. As emphasized by the Office Action, the VMM of Nelson is run on the kernel. 03/17/2010 Office Action at 5.

If the kernel 600 of Nelson is considered to be the "operating system" of claim 1, then that still does not satisfy the requirement of the claim that the virtual machine monitor is interposed between the operating system and the hardware. In Fig. 1 of Nelson, it is apparent that the kernel is provided between the VMM 300 and the hardware 100.

There is no mapping of elements of Nelson that would satisfy the combination recited in claim 1. Therefore, it is clear that claim 1 and its dependent claims are allowable over Nelson.

Reversal of the final rejection of the above claims is respectfully requested.

2. Claim 3.

Claim 3 depends from claim 1, and is therefore allowable for at least the same reasons as claim 1. Moreover, claim 3 further recites:

> booting the virtual machine monitor on the computer hardware, **booting the operating system on the virtual machine monitor**, and devirtualizing the computer hardware before interposing the virtual machine monitor at runtime.

Note that claim 3 specifically recites booting the operating system on the virtual machine monitor. The rejection of claim 3 cited the Abstract and column 2, lines 1-7, of Nelson. Specifically, the Office Action pointed to the statement in the Abstract that the COS is used to

boot the system as a whole. 03/17/2010 Office Action at 6. The Office Action also pointed to the statement in the Abstract that the VM runs via a virtual machine monitor, which is installed to run on the kernel. *Id.*

Neither the COS nor the kernel is booted on the virtual machine monitor, as claimed. In fact, the VMM in Fig. 1 of Nelson is run on the kernel, not the other way around. Moreover, the COS of Nelson also is not booted on the VMM.

The cited column 2 passage of Nelson refers to the VMM running directly on the underlying hardware in conventional systems. The cited column 2 passage of Nelson also states that in other systems, host operating systems are interposed as a software layer between the VMM and the hardware. Providing the host operating system between the VMM and the hardware means that the host operating system would not be booted **on the VMM**, as recited in claim 3.

Claim 3 is therefore further allowable for the foregoing reasons.

Reversal of the final rejection of the above claim is respectfully requested.

3. Claim 8.

Claim 8 depends indirectly from claim 1, and is therefore allowable for at least the same reasons as claim 1. Moreover, claim 8 further recites:

> wherein the privileged instructions are caused to trap to the virtual machine monitor by using a kernel module of the operating system to reduce a privilege level of the operating system from a higher privilege level.

With respect to claim 8, the Examiner cited column 3, lines 5-16, of Nelson. 03/17/2010 Office Action at 9. The cited passage of Nelson refers to the COS initializing the computer, where the COS runs at a most privileged, system level. The kernel is loaded via the COS, and upon starting the kernel, the kernel displaces the COS from the system level and the kernel runs

at the system level. According to the cited column 3 passage of Nelson, requests for system resources are submitted via the kernel. There is no hint in this passage or anywhere else in Nelson, of a kernel module of an operating system to reduce a privilege level of the operating system from a higher privileged level.

Stated differently, there is no hint that the COS would be reduced from a higher privilege level to a lower privilege level, or that the kernel would be reduced from a higher privilege level to a lower privilege level.

Claim 8 is therefore further allowable for the foregoing reasons.

Reversal of the final rejection of the above claim is respectfully requested.

4. Claims 31-33, 35-38, 40, 52-56, 58, 60, 73, 74.

Independent claims 31 and 52, and their respective dependent claims, are allowable over Nelson for similar reasons as for claim 1 stated above.

Reversal of the final rejection of the above claims is respectfully requested.

5. Claim 34.

Claim 34 depends from claim 31, and is therefore allowable for at least the same reasons as claim 31. Moreover, claim 34 is further allowable for similar reasons as claim 8.

Reversal of the final rejection of the above claim is respectfully requested.

6. Claims 17, 39, 59.

Claims 17, 39, and 59 depend respectively from base claims 1, 31, and 52, and are therefore allowable for at least the same reasons as corresponding base claims.

Claim 17 further recites:

> wherein the operating system includes a **dual-mode driver** that performs direct hardware control in a **first mode** and communicates with a device driver of the virtual machine monitor in a **second mode**; and wherein interposing the virtual machine monitor on the I/O device includes:
>
> setting the dual-mode driver to the second mode; and
>
> redirecting I/O interrupts to interrupt handlers in the virtual machine monitor instead of to interrupt handlers in the operating system.

It is respectfully submitted that there is no operating system in Nelson that has a dual-mode driver that can operate in one of two modes recited in claim 17: a first mode in which the dual-mode driver performs direct hardware control, and a second mode in which the dual-mode driver communicates with a device driver of the VMM. Note that during operation of Nelson, the kernel is provided between the VMM and hardware. Therefore, the kernel has direct access to the hardware depicted in Fig. 1, and thus, there will be no driver in the kernel of Nelson that is a dual-mode driver changeable between the first mode and the second mode recited in claim 17. Clearly, there would be no driver in the kernel of Nelson that would operate in the second mode, where in the second mode, the dual-mode driver communicates with a device driver of the VMM. That is because the driver of the kernel would access the hardware directly, since the kernel is located at a layer below the VMM, and the kernel is directly connected to the hardware.

In view of the foregoing, claim 17 (and claims 39 and 59) are further allowable for the foregoing reasons.

Reversal of the final rejection of the above claims is respectfully requested.

7. Claims 19-28, 30, 41-47, 49, 51, 62-70, 72.

Independent claim 19 is also not anticipated by Nelson.

Claim 19 recites:

> using a virtual machine monitor and an operating system on virtualized computer hardware, the method comprising devirtualizing the virtualized computer hardware at runtime of a computer containing the virtualized computer hardware,

wherein runtime includes a period of execution in the computer after booting and before shutdown,

wherein devirtualizing the virtualized computer hardware comprises stopping the virtual machine monitor.

The rejection of claim 19 on pages 15-16 of the 03/17/2010 Office Action focuses on the teaching in Nelson that the kernel can be unloaded and removed from the computer. Nelson, 5:8-25. A further discussion of the unloading is provided in column 21 of Nelson. *Id.*, 21:20-31. However, although Nelson refers to unloading the kernel, there is no specific teaching in Nelson that the VMM of Nelson is also unloaded. Nelson simply states that as a result of the unloading, the interrupt and fault handling is restored from the kernel to the first operating system (COS). *Id.*, 5:18-25; 21:28-32. Thus, the argument made by the Examiner that unloading of the kernel would cause the virtual machine monitor to be unloaded does not find support in the teachings of Nelson.

Therefore, claim 19 and its dependent claims are allowable over Nelson.

Independent claims 41 and 62 and their dependent claims are similarly allowable over Nelson.

Reversal of the final rejection of the above claims is respectfully requested.

8. Claims 29, 50, 71.

Claims 29, 50, and 71 are dependent upon base claims 19, 41, and 52, and are therefore allowable for at least the same reasons as corresponding base claims. Moreover, claims 29, 50, and 71 are further allowable for similar reasons as claims 17, 39, and 59.

Reversal of the final rejection of the above claims is respectfully requested.

B. Claim 61 was rejected under 35 U.S.C. § 102(b) as anticipated by Bugnion (U.S. Patent No. 6,075,938).

1. Claim 61.

Independent claim 61 was rejected as purportedly anticipated by Bugnion. Note that claim 61 recites an I/O driver having first and second modes of operation, where the I/O driver is operable in the first mode to interface directly between the operating system and the I/O device, and the I/O driver is operable in the second mode to interface between the operating system and the corresponding I/O driver of the virtual machine monitor.

The Examiner cited the following passages of Bugnion as purportedly disclosing the claimed subject matter: column 8, lines 27-29; column 9, lines 65-67; column 11, lines 48-51; column 17, lines 14-28. 03/17/2010 Office Action at 53. The cited column 11 passage of Bugnion refers to handling hardware interrupts directly by the VMM through its own device drivers. The cited column 17 passage of Bugnion refers to disco's monitor call interface for reducing the complexity and overhead of accessing I/O devices. The cited column 17 passage also notes that the monitor call interface provides a view of an idealized device, and the implementation of drivers is straight forward.

The cited column 8 passage of Bugnion refers to a multiprocessor that consists of a collection of nodes each containing a processor, main memory, and I/O devices. The cited column 9 passage of Bugnion refers to processors and physical memory, with operating systems assuming exclusive access to their I/O devices.

None of the passages of Bugnion provide any hint of an I/O driver that is operable in two modes of operation in the manner recited in claim 61. Therefore, claim 61 is clearly not anticipated by Bugnion.

Reversal of the final rejection of the above claim is respectfully requested.

CONCLUSION

In view of the foregoing, reversal of all final rejections and allowance of all pending claims is respectfully requested.

Respectfully submitted,

Date: August 16, 2010

/Dan C. Hu/

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
Telephone: (713) 468-8880
Facsimile: (713) 468-8883

VIII. APPENDIX OF APPEALED CLAIMS

Claims 2, 48, and 57 have been cancelled.

The claims on appeal are:

1. A method of using a virtual machine monitor and an operating system on computer hardware in a computer, the method comprising:

interposing the virtual machine monitor between the computer hardware and the operating system at runtime, wherein the interposing occurs after booting of the computer, and wherein interposing the virtual machine monitor gives the virtual machine monitor direct control of at least a portion of the computer hardware; and

booting the operating system on the computer hardware before interposing the virtual machine monitor at runtime.

3. The method of claim 1, further comprising booting the virtual machine monitor on the computer hardware, booting the operating system on the virtual machine monitor, and devirtualizing the computer hardware before interposing the virtual machine monitor at runtime.

4. The method of claim 1, further comprising devirtualizing the computer hardware at runtime after the virtual machine monitor has been interposed.

5. The method of claim 1, wherein the computer hardware includes a CPU; and wherein the virtual machine monitor is interposed on the CPU.

6. The method of claim 5, wherein the computer hardware further includes memory, and the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein interposing the virtual machine monitor on the CPU includes:

causing privileged instructions to trap to the virtual machine monitor, and

redirecting interrupts to the corresponding virtual machine monitor CPU interrupt handlers instead of to the operating system CPU interrupt handlers.

7. The method of claim 6, wherein the privileged instructions are caused to trap to the virtual machine monitor by causing the operating system to run at a reduced privilege level; and wherein interposing the virtual machine monitor on the CPU further includes returning control to the operating system at the reduced privilege level.

8. The method of claim 6, wherein the privileged instructions are caused to trap to the virtual machine monitor by using a kernel module of the operating system to reduce a privilege level of the operating system from a higher privilege level.

9. The method of claim 6, wherein interposing the virtual machine monitor on the CPU further includes disabling physical memory access by the operating system.

10. The method of claim 6, wherein interposing the virtual machine monitor on the CPU further includes loading the virtual machine monitor into the memory.

11. The method of claim 10, further comprising using a kernel module of the operating system to allocate memory within the operating system, pin the allocated memory, and load the virtual machine monitor into the pinned memory.

12. The method of claim 5, wherein the computer hardware includes memory; and wherein the virtual machine monitor is also interposed on the memory.

13. The method of claim 12, wherein interposing the virtual machine monitor on the memory includes partitioning the memory to provide partitions, and giving the virtual machine monitor access to at least one of the partitions.

14. The method of claim 12, wherein interposing the virtual machine monitor on the memory includes using a kernel module of the operating system to allocate a block of the memory, pin the block to prevent the operating system from using the block, and allocate the pinned block to the virtual machine monitor.

15. The method of claim 12, wherein interposing the virtual machine monitor on the memory includes commencing using the virtual machine monitor at runtime to manage memory translation.

16. The method of claim 5, wherein the computer hardware includes an I/O device, and wherein the virtual machine monitor is also interposed on the I/O device.

17. The method of claim 16, wherein the operating system includes a dual-mode driver that performs direct hardware control in a first mode and communicates with a device driver of the virtual machine monitor in a second mode; and wherein interposing the virtual machine monitor on the I/O device includes:

setting the dual-mode driver to the second mode; and

redirecting I/O interrupts to interrupt handlers in the virtual machine monitor instead of to interrupt handlers in the operating system.

18. The method of claim 16, wherein interposing the virtual machine monitor on the I/O device includes commencing I/O emulation of the I/O device at runtime.

19. A method of using a virtual machine monitor and an operating system on virtualized computer hardware, the method comprising devirtualizing the virtualized computer hardware at runtime of a computer containing the virtualized computer hardware, wherein runtime includes a period of execution in the computer after booting and before shutdown,

wherein devirtualizing the virtualized computer hardware comprises stopping the virtual machine monitor.

20. The method of claim 19, wherein the virtualized computer hardware includes a CPU; and wherein the CPU is devirtualized at runtime.

21. The method of claim 20, wherein the virtualized computer hardware further includes physical memory, and the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein devirtualizing the CPU includes redirecting interrupts to the corresponding operating system CPU interrupt handlers instead of to the virtual machine monitor CPU interrupt handlers.

22. The method of claim 21, wherein devirtualizing the CPU further includes restoring a privilege level of the operating system from a less privileged mode to a more privileged mode.

23. The method of claim 21, wherein devirtualizing the CPU further includes enabling physical memory access by the operating system.

24. The method of claim 21, wherein devirtualizing the CPU further includes unloading the virtual machine monitor from the physical memory.

25. The method of claim 19, wherein the virtualized computer hardware includes memory; and wherein the memory is devirtualized at runtime.

26. The method of claim 25, wherein memory was allocated from the operating system to the virtual machine monitor during virtualization of the memory; and wherein devirtualizing the memory includes returning the allocated memory to the operating system.

27. The method of claim 25, wherein devirtualizing the memory includes remapping physical memory and using the operating system to manage address translation with respect to the devirtualized memory.

28. The method of claim 19, wherein the virtualized computer hardware includes an I/O device, and wherein the I/O device is devirtualized at runtime.

29. The method of claim 28, wherein the operating system includes a dual-mode driver that performs direct hardware control in a first mode and communicates with a device driver of the virtual machine monitor in a second mode; and wherein devirtualizing the I/O device includes:

setting the dual-mode driver to the first mode from the second mode, and

redirecting I/O interrupts to handlers in the operating system instead of handlers in the virtual machine monitor.

30. The method of claim 28, wherein devirtualizing the I/O device includes ceasing emulation of the I/O device at runtime.

31. A computer comprising hardware, the hardware including memory, the memory encoded with an operating system, a virtual machine monitor, and code for interposing the virtual machine monitor between the operating system and the hardware at runtime, wherein the interposing occurs after booting of the computer,

wherein the operating system is to be booted in the computer before interposing the virtual machine monitor.

32. The computer of claim 31, wherein the hardware further includes a CPU, wherein the virtual machine monitor is interposed on the CPU at runtime, and the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein the interposing code is to cause privileged instructions to trap to the virtual machine monitor, and to redirect interrupts and traps to the corresponding virtual machine monitor CPU interrupt handlers instead of to the operating system CPU interrupt handlers.

33. The computer of claim 32, wherein the interposing code is to cause privileged instructions to trap to the virtual machine monitor by causing the operating system to run at a reduced privilege level from a higher privilege level; and wherein the interposing code is to reduce a privilege level of the operating system after redirecting the interrupts, and to return control to the operating system at the reduced privilege level.

34. The computer of claim 32, wherein the interposing code includes a kernel module of the operating system for reducing a privilege level of the operating system from a higher privilege level, whereby the privileged instructions trap to the virtual machine monitor.

35. The computer of claim 32, wherein the interposing code is to disable physical memory access by the operating system.

36. The computer of claim 31, wherein the interposing code includes a kernel module of the operating system for allocating a block of the memory, pinning the block to prevent the operating system from using the block, and allocating the pinned block to the virtual machine monitor, whereby the virtual machine monitor is interposed on the memory at runtime.

37. The computer claim 31, wherein the interposing code is to commence using the virtual machine monitor at runtime to manage memory translation, whereby the virtual machine monitor is interposed on the memory at runtime.

38. The computer of claim 31, wherein the hardware further includes an I/O device; and wherein the interposing code includes an operating system dual-mode driver to perform direct hardware control in a first mode and to communicate with a device driver of the virtual machine monitor in a second mode; and wherein the interposing code is to set the dual-mode driver to the second mode, and to direct I/O interrupts to interrupt handlers in the virtual machine monitor instead of to interrupt handlers in the operating system, whereby the virtual machine monitor is interposed on the I/O device at runtime.

39. The computer of claim 31, wherein the hardware further includes an I/O device; and wherein the operating system includes a dual-mode driver to perform direct hardware control in a first mode and to communicate with a device driver of the virtual machine monitor in a second mode; and wherein the interposing code is to set the dual-mode driver to the second mode, and to redirect I/O interrupts to interrupt handlers in the virtual machine monitor instead of to interrupt handlers in the operating system, whereby the virtual machine monitor is interposed on the I/O device.

40. The computer of claim 31, wherein the hardware further includes an I/O device; and wherein the interposing code is to commence I/O emulation of the I/O device at runtime, whereby the virtual machine monitor is interposed on the I/O device at runtime.

41. A computer comprising hardware, the hardware including memory, the memory encoded with a virtual machine monitor to virtualize the hardware, and code for devirtualizing the hardware at runtime, wherein runtime includes a period of execution in the computer after booting and before shutdown, and wherein devirtualizing the hardware comprises stopping the virtual machine monitor.

42. The computer of claim 41, wherein the hardware further includes a CPU; and wherein the devirtualizing code is to devirtualize the CPU at runtime.

43. The computer of claim 42, wherein the memory is further encoded with an operating system including interrupt handlers; wherein the virtual machine monitor includes interrupt handlers; and wherein the devirtualizing code is to redirect interrupts to the corresponding interrupt handlers of the operating system instead of to the interrupt handlers of the virtual machine monitor.

44. The computer of claim 43, wherein the devirtualizing code is to restore privilege level of the operating system from a lower privilege level to a higher privilege level.

45. The computer of claim 43, wherein the devirtualizing code is to enable physical memory access by the operating system.

46. The computer of claim 41, wherein the devirtualizing code is to devirtualize the memory at runtime.

47. The computer of claim 46, wherein the virtual machine monitor is to allocate memory from an operating system to the virtual machine monitor; and wherein the devirtualizing code is to return the allocated memory to the operating system.

49. The computer of claim 41, wherein the hardware includes an I/O device, wherein the virtual machine monitor is to virtualize the I/O device; and wherein the devirtualizing code is to devirtualize the I/O device at runtime.

50. The computer of claim 49, wherein the memory is further encoded with an operating system including dual-mode drivers to perform direct hardware control in a first mode and communicate with device drivers of the virtual machine monitor in a second mode; and wherein the devirtualizing code is to set the dual-mode drivers to the first mode from the second mode, and to redirect I/O interrupts to handlers in the operating system instead of to handlers in the virtual machine monitor.

51. The computer of claim 49, wherein the devirtualizing code is to cease emulation of the I/O device at runtime.

52. An article for use with an operating system on computer hardware, the article comprising a computer-readable storage medium storing software that when executed by the computer causes the computer to:

virtualize at least a portion of the computer hardware at runtime by providing a virtual machine monitor between the operating system and the computer hardware, wherein the virtualizing occurs after booting of the computer and loading of the operating system, and

wherein the operating system is to be booted in the computer before virtualizing the at least a portion of the computer hardware at runtime.

53. The article of claim 52, wherein the computer hardware further includes a CPU, and wherein the virtual machine monitor and the operating system each include CPU interrupt handlers; and wherein the software is executable to cause privileged instructions to trap to the virtual machine monitor, and to cause interrupts and traps to be redirected to the corresponding virtual machine monitor interrupt handlers instead of to the operating system interrupt handlers.

54. The article of claim 53, wherein the software is executable to cause the privileged instructions to trap to the virtual machine monitor by reducing a privilege level of the operating system from a higher privilege level, and wherein the software causes control to be returned to the operating system at the reduced privilege level.

55. The article of claim 53, wherein the software is executable to cause physical memory access by the operating system to be disabled.

56. The article of claim 52, wherein the computer hardware includes memory, and wherein the virtual machine monitor is for causing a kernel module of the operating system to allocate a block of the memory, pin the block to prevent the operating system from using the block, and allocate the pinned block to the virtual machine monitor.

58. The article of claim 52, wherein the computer hardware further includes an I/O device; and wherein the software includes an operating system dual-mode driver to perform direct hardware control in a first mode and communicate with a corresponding device driver of a virtual machine monitor in a second mode; and wherein the dual-mode driver is set to the second mode when the at least the portion of the computer hardware is virtualized, and wherein I/O interrupts are redirected to interrupt handlers in the virtual machine monitor instead of interrupt handlers in the operating system.

59. The article of claim 52, wherein the computer hardware further includes an I/O device; and wherein the operating system includes a dual-mode driver to perform direct hardware control in a first mode and communicate with a device driver of the virtual machine monitor in a second mode; and wherein the dual-mode driver is set to the second mode when the at least the portion of the computer hardware is virtualized, and wherein I/O interrupts are redirected from interrupt handlers in the operating system to interrupt handlers in the virtual machine monitor.

60. The article of claim 52, wherein the computer hardware further includes an I/O device; and wherein the software is executable to cause I/O emulation of the I/O device to commence at runtime.

61. An article for running an operating system and a virtual machine monitor on a computer, the computer including an I/O device, the article comprising computer memory encoded with an I/O driver having first and second modes of operation, the I/O driver operable in the first mode to interface directly between the operating system and the I/O device, the I/O driver operable in the second mode to interface between the operating system and a corresponding I/O driver of the virtual machine monitor.

62. An article for use with an operating system on computer hardware, the article comprising a computer-readable storage medium storing software that when executed by a computer causes the computer to devirtualize at least a portion of virtualized hardware at runtime, wherein runtime is a period of execution in the computer after booting and before shutdown, and wherein devirtualizing the at least a portion of the virtualized hardware comprises stopping a virtual machine monitor interposed between the operating system and the hardware.

63. The article of claim 62, wherein the virtualized hardware includes a CPU; and wherein the software causes the CPU to be devirtualized at runtime.

64. The article of claim 63, wherein the virtualized hardware further includes memory, and wherein the memory is further encoded with the operating system including first interrupt handlers; wherein the software includes second interrupt handlers; and wherein the software is executable to cause interrupts to be redirected to the corresponding first interrupt handlers instead of to the second interrupt handlers.

65. The article of claim 64, wherein the software is executable to cause a privilege level of the operating system to be restored from a lower privilege level to a higher privilege level.

66. The article of claim 64, wherein the software is executable to cause physical memory access by the operating system to be enabled.

67. The article of claim 62, wherein the virtualized hardware includes a memory, and wherein the software is executable to cause the memory to be devirtualized at runtime.

68. The article of claim 67, wherein if a part of the memory was allocated from an operating system to the virtual machine monitor prior to the runtime devirtualization, the software is executable to cause the allocated memory to be returned to the operating system as part of the runtime devirtualization.

69. The article of claim 67, wherein the software is executable to cause physical memory to be remapped and wherein the software allows an operating system to manage address translation with respect to the devirtualized memory.

70. The article of claim 62, wherein the virtualized hardware includes an I/O device; and wherein the software is executable to cause the I/O device to be devirtualized at runtime.

71. The article of claim 70, wherein the virtualized hardware further includes a memory, and wherein the memory is further encoded with the operating system including dual-mode drivers that perform direct hardware control in a first mode and communicate with virtual device drivers in a second mode; and wherein the software is executable to cause the dual-mode drivers to be set to the first mode.

72. The article of claim 70, wherein the software is executable to cause emulation of the I/O device to cease at runtime.

73. The computer of claim 31, wherein interposing the virtual machine monitor gives the virtual machine monitor direct control of at least a portion of the hardware such that the operating system no longer has direct control of the at least a portion of the hardware.

74. The article of claim 52, wherein providing the virtual machine monitor between the operating system and the computer hardware gives the virtual machine monitor direct control of at least a portion of the hardware such that the operating system no longer has direct control of the at least a portion of the hardware.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

The following related applications are under appeal:

U.S. Serial No.	Status
10/677,159	Appeal Brief filed 05/17/2010 Notice of Appeal filed 03/15/2010 Appeal Brief filed 02/05/2009 Notice of Appeal filed 12/05/2008
10/676,922	Appeal Brief filed 04/26/2010 Notice of Appeal filed 02/25/2010 Appeal Brief filed 01/09/2009 Notice of Appeal filed 11/11/2008 Appeal Brief filed 05/27/2008 Notice of Appeal filed 03/26/2008

No Decisions on Appeal have been rendered by the Board in any of these appeals.